Программирование

Python

Лекция 10

Кафедра ИВТ и ПМ
ЗабГУ

2018

План

Outline

Outline

Язык программирования

- Какие компоненты задают язык программирования?

Язык программирования

- Какие компоненты задают язык программирования?
  - Алфавит
  - Синтаксис
  - Семантика
- Что входит в **алфавит языка Python**?

Язык программирования

- Какие компоненты задают язык программирования?
  - Алфавит
  - Синтаксис
  - Семантика
- Что входит в **алфавит языка Python**?
  - Буквы национальных алфавитов, A-z, А-я, ...
  - Символ подчёркивания _
  - Цифры 0-9
  - Специальные символы + - * / > < =; ‘ , . : [] ( ) @
  - Комбинации символов (считаются одним символом, например: +=, while, and)

Лексемы

- Что такое **лексема**?

Лексемы

- Что такое **лексема**?
  последовательность допустимых символов языка программирования, имеющая смысл для транслятора
- Примеры лексем?

Лексемы

- Что такое **лексема**?
  последовательность допустимых символов языка программирования, имеющая смысл для транслятора
- Примеры лексем?

```
import
if
"qwerty"
123.74
print
x   # x - объявленная переменная
and
+


# комментарий не является лексемой
```

Лексемы

Виды лексем

- Идентификаторы (имена, identifiers)

```
pow  # идентификатор функции возведения в степень
my_var  # идентификатор переменной
math  # идентификатор модуля
```

- Служебные (зарезервированные слова, keywords)

```
if
import
in
pass
```

- Литералы (неименованные константы, literals)
- Знаки операций (punctuators)

Литерал

- Что такое **литерал**?

Литерал

- Что такое **литерал**?
  запись в исходном коде компьютерной программы, представляющая собой фиксированное значение

  Примеры:

```
123.8
"qwerty"
"z"
'also string'
False
-7
None
""
[]
```

Переменная

- Что такое **переменная**?

Переменная

- Что такое **переменная**?
  Переменная (в Python) - имя (идентификатор), с которым может быть связано значение.

- Как объявить переменную?

Переменная

- Что такое **переменная**?
  Переменная (в Python) - имя (идентификатор), с которым может быть связано значение.

- Как объявить переменную? Чтобы объявить переменную в Python необходимо задать ей значение

```
x = 0
```

```
A = None
```

Выражение

- Что такое **выражение**?

Выражение

- Что такое **выражение**?
  комбинация значений, констант, переменных, операций и функций, которая может быть интерпретирована в соответствии с правилами конкретного языка

Примеры выражений:

```
2
2**9
x  # x - переменная
x + 1
y + x
sin(0)
cos(x + 2)
abs( x - 2 ) + 8
a and b
x == 8
y > 0 > z
```

Оператор присваивания

- Что такое **оператор присваивания**?

Оператор присваивания

- Что такое **оператор присваивания**?
  оператор связывающий имя переменной с некоторым значением

Примеры:

```
# справа от оператора = может быть литерал
x = 10
s = 'some string'

# справа от оператора = может быть другая переменная
y = x

# справа от оператора = может быть выражение
x = 2 + 2
y = sin( x ) + 0.5
a = b and c
A = x==y
```

Outline

Типы данных

- Что такое **тип данных**?
  - множество допустимых значений, которые могут принимать данные, принадлежащие к этому типу;
  - набор операций, которые можно осуществлять над данными, принадлежащими к этому типу.

Типы данных

Простые типы данных Python:

- **bool** - логический
- **int** - целый
- **float** - вещественный
- **NoneType** - пустой

Составные типы данных

- **str** - строковый (набор символов)
- **tuple** - кортеж (неизменяемая линейная коллекция)
- **list** - список (изменяемая коллекция)
- **set** - множество (неупорядоченная коллекция уникальных значений)
- **dict** - словарь (неупорядоченная коллекция пар ключ-значение)

Использование функций

Что такое **функция**?

Использование функций

Что такое **функция**?
фрагмент программного кода (подпрограмма), к которому можно обратиться из другого места программы.

Как обратится к функции (вызвать)?

Использование функций

Что такое **функция**?
фрагмент программного кода (подпрограмма), к которому можно обратиться из другого места программы.

Как обратится к функции (вызвать)?

При обращении к функции (вызове) указывается её *имя* (идентификатор) и *параметры* в круглых скобках:

```
имя_функции( параметр1, параметр2, параметр3, ...)
```

У функции может не быть параметров.

Использование функций

Примеры вызовов функций:

```
exit()  # выход из программы

print("Hello World")  # вывод на экран

sleep( 5.16 )  # приостановка программы на 5.16 секунд

help( exit )  # вызов справки по указанному идентификат
```

Использование функций

Как определить число и тип параметров функции?

Использование функций

Как определить число и тип параметров функции?
Вызвав справки по функции

```
help( sleep )
```

```
    sleep(seconds)

    Delay execution for a given number of seconds.
    The argument may be a floating point number
    for subsecond precision.
```

Сколько параметров принимает функция?

Использование функций

Как определить число и тип параметров функции?
Вызвав справки по функции

```
help( sleep )
```

```
    sleep(seconds)

    Delay execution for a given number of seconds.
    The argument may be a floating point number
    for subsecond precision.
```

Сколько параметров принимает функция? Один.

Какого типа должен быть параметр?

Использование функций

Как определить число и тип параметров функции?
Вызвав справки по функции

```
help( sleep )
```

```
    sleep(seconds)

    Delay execution for a given number of seconds.
    The argument may be a floating point number
    for subsecond precision.
```

Сколько параметров принимает функция? Один.

Какого типа должен быть параметр? Целого или вещественного.

Использование функций

Что означает фраза "функция возвращает значение"?

Использование функций

Что означает фраза "функция возвращает значение"?

Результат выполнения функции может быть записан в переменную:

Можно представлять, что после вызова и выполнения функции, на её место будет подставлен результат (значение функции).

Использование функций

Возвращает ли функция значение можно посмотреть в документации.

```
pow(x, y)

Return x**y (x to the power of y).
```

Что делает функция?

Использование функций

Возвращает ли функция значение можно посмотреть в документации.

```
pow(x, y)

Return x**y (x to the power of y).
```

Что делает функция?

Возводит x в степень y.

Функция pow возвращает результат?

Использование функций

Возвращает ли функция значение можно посмотреть в документации.

```
pow(x, y)

Return x**y (x to the power of y).
```

Что делает функция?

Возводит x в степень y.

Функция pow возвращает результат? - Да

Какого типа будет возвращаемое значение?

Использование функций

Возвращает ли функция значение можно посмотреть в документации.

```
pow(x, y)

Return x**y (x to the power of y).
```

Что делает функция?

Возводит x в степень y.

Функция pow возвращает результат? - Да

Какого типа будет возвращаемое значение? - Вещественного.

Если типы возвращаемого значения и параметров очевидны, то в документации они не указываются.

Прошлые темы

- Значение какого типа возвращает функция log?

Прошлые темы

- Значение какого типа возвращает функция log?
  float (вещественного)
- ... функция input?

Прошлые темы

- Значение какого типа возвращает функция log?
  float (вещественного)
- ... функция input?
  str (строкового)
- ... функция sleep?

Прошлые темы

- Значение какого типа возвращает функция log?
  float (вещественного)
- ... функция input?
  str (строкового)
- ... функция sleep?
  функция sleep не возвращает значения

Outline

Кодирование vs программирование

Кодирование - процесс написания программного кода.

Программирование = анализ + проектирование + кодирование + трансляция (компиляция, интерпретация) + тестирование + отладка ...

Стандарт оформления код

Стандарт оформления кода (стандарт кодирования, стиль программирования) (coding standards) — набор правил и соглашений, используемых при написании исходного кода на некотором языке программирования.

Стандарт оформления код

Стандарт оформления кода определяет:

- выбор имён идентификаторов;

  i,j,k,l - используется для обозначения индексов, счётчиков и т.п.
  N,n,m - обозначают количество

  Коллекции (списки, кортежи, словари) лучше всего называть в соответствии с их содержимым:

```
students = ["Шишкин", "Мышкин", "Пушкин"]
prices = [1000, 2050, 870, 40]   # цены в рубляx
points = [(-7,5), (1.2, 0), (2, -1.5)]   # точки на плос
```

  При этом идентификаторы должны быть лаконичны.

Стандарт оформления код

Стандарт оформления кода определяет:

- Использование переносов строки Логические блоки программы лучше отделять друг от друга одной или двумя пустыми строками. Например ввод данных, вычисление и вывод данных:

```
x = float(input("x = "))
y = float(input("y = "))

a = x + y
b = x % y

print("a = {:.2}".format(a))
print("b = {:.2}".format(b))
```

Стандарт оформления код

Стандарт оформления кода определяет:

- использование пробелов при оформлении логических и арифметических выражений

  операции стоит выделять пробелами, а в длинных выражениях пробелами стоит выделять отдельный блоки

```
a = sqrt( fabs(sin(8 * h)) + 17 \
        / (1 - sin(4 * h * (h**2 + 18)))**2 )
```

  Особенно длинные выражение нужно записывать в несколько строк, после каждого переноса строки ставя обратную косую черту.

Стандарт оформления код

Стандарт оформления кода определяет:

- стиль комментариев и использование документирующих комментариев

  Основное правило: комментарии нужно писать там, где код нельзя написать понятнее.

  Однако, во время изучения языка программирования (или модуля, библиотеки) рекомендуется использовать комментарии в широких пределах.

Outline

Теорема о структурном программировании

Любая программа, заданная в виде блок-схемы, может быть представлена с помощью трех управляющих структур:

- последовательность — обозначается: f THEN g,
- ветвление — обозначается: IF p THEN f ELSE g,
- цикл — обозначается: WHILE p DO f,

где f, g — блок-схемы с одним входом и одним выходом,

p — условие,
THEN, IF, ELSE, WHILE, DO — ключевые слова

Пояснение. Формула f THEN g означает следующее: сначала выполняется программа f, затем выполняется программа g.

Структурное программирование

- Программа состоит из трёх базовых конструкций: последовательность, ветвление, цикл. Эти конструкции могут комбинироваться произвольным образом.
- Повторяющиеся фрагменты программы (или её логические блоки) оформляются как подпрограммы.
- Разработка ведётся "сверху-вниз": от общего алгоритма к более специальным.

Outline

Ссылки и литература

Ссылка на слайды

github.com/VetrovSV/Programming